2장 서울시 범죄 현황 분석(75-109쪽)

이름 : 구명회
학번 : 20154215
학과 : 컴퓨터 공학과

1. Pandas를 이용하여 데이터 정리하기
2. Google Maps 이용하여 지도 활용하기 (x)
3. Pivot_table을 이용한 데이터 정리
4. Seaborn을 이용한 시각화
5. 범죄데이터 시각화
6. Folium을 이용한 지도 시각화 (x)
7. 범죄율에 대한 지도 시각화 (x)
8. 경찰서별 검거율과 구별 범죄율 시각화 (x)
9. 소감

Python Pandas

1. Pandas를 이용하여 데이터 정리하기

Working With Text Data

In [1]:
```
# numpy와 pandas 모듈을 import
import numpy as np_hw2
import pandas as pd_hw2
```

In [2]:
```
# data를 pandas로 읽어서 crime_hw2 변수에 저장합니다.
# .read_csv : CSV 파일을 읽는 명령, encoding = 'euc-kr' : 인코딩 data를 euc-kr을 사용하겠다
# euc-kr : 2바이트로 한글이나 일부 한자 등을 표현할 수 있게 만든 방식. 아스키값은 그대로 1바이트
# .head() : 인자값만큼 행을 출력한다. 인자를 default로 주면 pandas data의 첫 5행만 출력
crime_hw2 = pd_hw2.read_csv('../data/02. crime_in_Seoul.csv', thousands=',', encoding='euc-kr')
crime_hw2.head()
```

Out[2]:

	관서명	살인 발생	살인 검거	강도 발생	강도 검거	강간 발생	강간 검거	절도 발생	절도 검거	폭력 발생	폭력 검거
0	중부서	2	2	3	2	105	65	1395	477	1355	1170
1	종로서	3	3	6	5	115	98	1070	413	1278	1070
2	남대문서	1	0	6	4	65	46	1153	382	869	794
3	서대문서	2	2	5	4	154	124	1812	738	2056	1711
4	혜화서	3	2	5	4	96	63	1114	424	1015	861

2. Google Maps 이용하여 지도 활용하기

주말에 따로 api 받아서 실습해보고 ppt에 추가하겠습니다.

3. Pivot_table을 이용한 데이터 정리

3-1 Pivot table 사용해서 데이터 학습하기
3-2 Pivot table 사용해서 데이터 정리하기
3-3 데이터 표현을 위해 다듬기

3-1 Pivot table 사용해서 데이터 학습하기

예제의 import문을 삭제하고
1. 에서 사용한 pandas와 numpy모듈 변수를 그대로 사용했습니다.

In [3]:
```
# data를 pandas로 읽어서 df_hw2 변수에 저장합니다.
# .read_excel : excel 파일을 읽는 명령
# .head() : 인자값만큼 행을 출력한다. 인자를 default로 주면 pandas data의 첫 5행만 출력
df_hw2 = pd_hw2.read_excel("../data/02. sales-funnel.xlsx")
df_hw2.head()
```

Out [3]:

	Account	Name	Rep	Manager	Product	Quantity	Price	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

In [4]:
```
# index를 Name항목으로만 정렬합니다.
# 중복되는 Name 항목은 하나로 합쳐지고 value들은 평균을 갖음
pd_hw2.pivot_table(df_hw2,index=["Name"])
```

Out [4]:

Name	Account	Price	Quantity
Barton LLC	740150	35000	1.000000
Fritsch, Russel and Anderson	737550	35000	1.000000
Herman LLC	141962	65000	2.000000
Jerde-Hilpert	412290	5000	2.000000
Kassulke, Ondricka and Metz	307599	7000	3.000000
Keeling LLC	688981	100000	5.000000
Kiehn-Spinka	146832	65000	2.000000
Koepp Ltd	729833	35000	2.000000
Kulas Inc	218895	25000	1.500000
Purdy-Kunde	163416	30000	1.000000
Stokes LLC	239344	7500	1.000000
Trantow-Barrows	714466	15000	1.333333

In [5]:
```
# index를 여러 개 지정할 수 있습니다.
pd_hw2.pivot_table(df_hw2,index=["Name","Rep","Manager"])
```

Out [5]:

Name	Rep	Manager	Account	Price	Quantity
Barton LLC	John Smith	Debra Henley	740150	35000	1.000000
Fritsch, Russel and Anderson	Craig Booker	Debra Henley	737550	35000	1.000000
Herman LLC	Cedric Moss	Fred Anderson	141962	65000	2.000000
Jerde-Hilpert	John Smith	Debra Henley	412290	5000	2.000000
Kassulke, Ondricka and Metz	Wendy Yule	Fred Anderson	307599	7000	3.000000
Keeling LLC	Wendy Yule	Fred Anderson	688981	100000	5.000000
Kiehn-Spinka	Daniel Hilton	Debra Henley	146832	65000	2.000000
Koepp Ltd	Wendy Yule	Fred Anderson	729833	35000	2.000000
Kulas Inc	Daniel Hilton	Debra Henley	218895	25000	1.500000
Purdy-Kunde	Cedric Moss	Fred Anderson	163416	30000	1.000000
Stokes LLC	Cedric Moss	Fred Anderson	239344	7500	1.000000
Trantow-Barrows	Craig Booker	Debra Henley	714466	15000	1.333333

In [6]:
```
# 예제
pd_hw2.pivot_table(df_hw2,index=["Manager","Rep"])
```

Out[6]:

Manager	Rep	Account	Price	Quantity
Debra Henley	Craig Booker	720237.0	20000.000000	1.250000
	Daniel Hilton	194874.0	38333.333333	1.666667
	John Smith	576220.0	20000.000000	1.500000
Fred Anderson	Cedric Moss	196016.5	27500.000000	1.250000
	Wendy Yule	614061.5	44250.000000	3.000000

In [7]:
```
# 특정 value를 지정해서 출력할 수 있습니다.
pd_hw2.pivot_table(df_hw2,
      index=["Manager","Rep"], values=["Price"])
```

Out[7]:

		Price
Manager	Rep	
Debra Henley	Craig Booker	20000.000000
	Daniel Hilton	38333.333333
	John Smith	20000.000000
Fred Anderson	Cedric Moss	27500.000000
	Wendy Yule	44250.000000

In [8]:
```
# value를 pivot_table로 합치는 경우 평균치가 기본이 됩니다.
# 합계를 사용하려면 aggfunc옵션에 numpy.sum을 사용합니다.
pd_hw2.pivot_table(df_hw2,index=["Manager","Rep"],
                   values=["Price"],aggfunc=np_hw2.sum)
```

Out[8]:

		Price
Manager	Rep	
Debra Henley	Craig Booker	80000
	Daniel Hilton	115000
	John Smith	40000
Fred Anderson	Cedric Moss	110000
	Wendy Yule	177000

In [9]:
```
# 예제
pd_hw2.pivot_table(df_hw2,index=
                   ["Manager","Rep"],values=["Price"],
                  aggfunc=[np_hw2.mean,len])
```

Out[9]:

		mean	len
		Price	Price
Manager	Rep		
Debra Henley	Craig Booker	20000.000000	4
	Daniel Hilton	38333.333333	3
	John Smith	20000.000000	2
Fred Anderson	Cedric Moss	27500.000000	4
	Wendy Yule	44250.000000	4

In [10]:
```
# 예제
pd_hw2.pivot_table(df_hw2,index=["Manager","Rep"],values=["Price"],
                   columns=["Product"],aggfunc=[np_hw2.sum])
```

Out[10]:

		sum			
		Price			
	Product	CPU	Maintenance	Monitor	Software
Manager	Rep				
Debra Henley	Craig Booker	65000.0	5000.0	NaN	10000.0
	Daniel Hilton	105000.0	NaN	NaN	10000.0
	John Smith	35000.0	5000.0	NaN	NaN
Fred Anderson	Cedric Moss	95000.0	5000.0	NaN	10000.0
	Wendy Yule	165000.0	7000.0	5000.0	NaN

In [11]:
```
# 예제
# fill_value = 0 : nan을 0으로 채웁니다.
pd_hw2.pivot_table(df_hw2,index=["Manager","Rep"],values=["Price"],
                   columns=["Product"],aggfunc=[np_hw2.sum],fill_value=0)
```

Out[11]:

		sum			
		Price			
	Product	CPU	Maintenance	Monitor	Software
Manager	Rep				
Debra Henley	Craig Booker	65000	5000	0	10000
	Daniel Hilton	105000	0	0	10000
	John Smith	35000	5000	0	0
Fred Anderson	Cedric Moss	95000	5000	0	10000
	Wendy Yule	165000	7000	5000	0

In [12]:
```
# 예제
pd_hw2.pivot_table(df_hw2,
                   index=["Manager","Rep","Product"],
                   values=["Price","Quantity"],
                   aggfunc=[np_hw2.sum],fill_value=0)
```

Out[12]:

			sum	
			Price	Quantity
Manager	Rep	Product		
Debra Henley	Craig Booker	CPU	65000	2
		Maintenance	5000	2
		Software	10000	1
	Daniel Hilton	CPU	105000	4
		Software	10000	1
	John Smith	CPU	35000	1
		Maintenance	5000	2
Fred Anderson	Cedric Moss	CPU	95000	3
		Maintenance	5000	1
		Software	10000	1
	Wendy Yule	CPU	165000	7
		Maintenance	7000	3
		Monitor	5000	2

In [13]:
```
# Index와 value를 지정합니다.
# 합산과 평균을 numpy.sum과 numpy.mean으로 표시합니다.
# 빈칸은 fill_value옵션을 사용해서 0으로 채웁니다.
pd_hw2.pivot_table(df_hw2,index=["Manager","Rep","Product"],
                   values=["Price","Quantity"],
                   aggfunc=[np_hw2.sum,np_hw2.mean],fill_value=0,margins=True)
```

Out[13]:

			sum		mean	
			Price	Quantity	Price	Quantity
Manager	Rep	Product				
Debra Henley	Craig Booker	CPU	65000	2	32500	1.000000
		Maintenance	5000	2	5000	2.000000
		Software	10000	1	10000	1.000000
	Daniel Hilton	CPU	105000	4	52500	2.000000
		Software	10000	1	10000	1.000000
	John Smith	CPU	35000	1	35000	1.000000
		Maintenance	5000	2	5000	2.000000
Fred Anderson	Cedric Moss	CPU	95000	3	47500	1.500000
		Maintenance	5000	1	5000	1.000000
		Software	10000	1	10000	1.000000
	Wendy Yule	CPU	165000	7	82500	3.500000
		Maintenance	7000	3	7000	3.000000
		Monitor	5000	2	5000	2.000000
All			522000	30	30705	1.764706

3-2 Pivot table 사용해서 데이터 정리하기

예제의 import문을 삭제하고
1. 에서 사용한 pandas와 numpy모듈 변수를 그대로 사용했습니다.

In [14]:
```
# data를 pandas로 읽어서 crime_anal_hw2 변수에 저장합니다.
# .read_csv : CSV 파일을 읽는 명령, encoding = 'utf-8' : 인코딩 data를 utf-8을 사용하겠다
# utf-8 : 가변길이 유니코드 인코딩, 한 글자당 1~4바이트 사이의 크기로 인코딩
# .head() : 인자값만큼 행을 출력한다. 인자를 default로 주면 pandas data의 첫 5행만 출력
crime_anal_hw2 = pd_hw2.read_csv('../data/02. crime_in_Seoul_include_gu_name.csv', encoding='utf-8')
crime_anal_hw2.head()
```

Out[14]:

	Unnamed: 0	관서명	살인 발생	살인 검거	강도 발생	강도 검거	강간 발생	강간 검거	절도 발생	절도 검거	폭력 발생	폭력 검거	구별
0	0	중부서	2	2	3	2	105	65	1395	477	1355	1170	중구
1	1	종로서	3	3	6	5	115	98	1070	413	1278	1070	종로구
2	2	남대문서	1	0	6	4	65	46	1153	382	869	794	중구
3	3	서대문서	2	2	5	4	154	124	1812	738	2056	1711	서대문구
4	4	혜화서	3	2	5	4	96	63	1114	424	1015	861	종로구

In [15]:
```
# 32line의 원데이터의 관서명을 구별로 바꾸었습니다.
# 0번째 열을 인덱스로 사용합니다.
crime_anal_hw2 = pd_hw2.read_csv('../data/02. crime_in_Seoul_include_gu_name.csv',
                                 encoding='utf-8', index_col=0)

crime_anal = pd_hw2.pivot_table(crime_anal_hw2, index='구별', aggfunc=np_hw2.sum)
crime_anal.head()
```

Out [15]:

구별	강간 검거	강간 발생	강도 검거	강도 발생	살인 검거	살인 발생	절도 검거	절도 발생	폭력 검거	폭력 발생
강남구	349	449	18	21	10	13	1650	3850	3705	4284
강동구	123	156	8	6	3	4	789	2366	2248	2712
강북구	126	153	13	14	8	7	618	1434	2348	2649
관악구	221	320	14	12	8	9	827	2706	2642	3298
광진구	220	240	26	14	4	4	1277	3026	2180	2625

In [16]:
```
# 각 범죄별 검거율을 계산합니다.
crime_anal['강간검거율'] = crime_anal['강간 검거']/crime_anal['강간 발생']*100
crime_anal['강도검거율'] = crime_anal['강도 검거']/crime_anal['강도 발생']*100
crime_anal['살인검거율'] = crime_anal['살인 검거']/crime_anal['살인 발생']*100
crime_anal['절도검거율'] = crime_anal['절도 검거']/crime_anal['절도 발생']*100
crime_anal['폭력검거율'] = crime_anal['폭력 검거']/crime_anal['폭력 발생']*100

# 각 범죄별 검거 열을 삭제합니다.
del crime_anal['강간 검거']
del crime_anal['강도 검거']
del crime_anal['살인 검거']
del crime_anal['절도 검거']
del crime_anal['폭력 검거']

crime_anal.head()
```

Out[16]:

구별	강간 발생	강도 발생	살인 발생	절도 발생	폭력 발생	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율
강남구	449	21	13	3850	4284	77.728285	85.714286	76.923077	42.857143	86.484594
강동구	156	6	4	2366	2712	78.846154	133.333333	75.000000	33.347422	82.890855
강북구	153	14	7	1434	2649	82.352941	92.857143	114.285714	43.096234	88.637222
관악구	320	12	9	2706	3298	69.062500	116.666667	88.888889	30.561715	80.109157
광진구	240	14	4	3026	2625	91.666667	185.714286	100.000000	42.200925	83.047619

In [17]:
```
# 각 검거율중에서 100이상의 검거율은 100으로 수정합니다.
con_list_hw2 = ['강간검거율', '강도검거율', '살인검거율', '절도검거율', '폭력검거율']

for column in con_list_hw2:
    crime_anal.loc[crime_anal[column] > 100, column] = 100

crime_anal.head()
```

Out[17]:

구별	강간 발생	강도 발생	살인 발생	절도 발생	폭력 발생	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율
강남구	449	21	13	3850	4284	77.728285	85.714286	76.923077	42.857143	86.484594
강동구	156	6	4	2366	2712	78.846154	100.000000	75.000000	33.347422	82.890855
강북구	153	14	7	1434	2649	82.352941	92.857143	100.000000	43.096234	88.637222
관악구	320	12	9	2706	3298	69.062500	100.000000	88.888889	30.561715	80.109157
광진구	240	14	4	3026	2625	91.666667	100.000000	100.000000	42.200925	83.047619

In [18]:
```
# .rename을 사용해서 단어를 제거합니다.
crime_anal.rename(columns = {'강간 발생':'강간',
                             '강도 발생':'강도',
                             '살인 발생':'살인',
                             '절도 발생':'절도',
                             '폭력 발생':'폭력'}, inplace=True)
crime_anal.head()
```

Out[18]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율
강남구	449	21	13	3850	4284	77.728285	85.714286	76.923077	42.857143	86.484594
강동구	156	6	4	2366	2712	78.846154	100.000000	75.000000	33.347422	82.890855
강북구	153	14	7	1434	2649	82.352941	92.857143	100.000000	43.096234	88.637222
관악구	320	12	9	2706	3298	69.062500	100.000000	88.888889	30.561715	80.109157
광진구	240	14	4	3026	2625	91.666667	100.000000	100.000000	42.200925	83.047619

3-3 데이터 표현을 위해 다듬기

In [19]:
```
# 정규화
from sklearn import preprocessing

# 각 범죄마다 건수가 차이가 나기 때문에
# 정규화하여 0~1의 값을 가지도록 합니다.
col = ['강간', '강도', '살인', '절도', '폭력']

x = crime_anal[col].values
min_max_scaler = preprocessing.MinMaxScaler()

x_scaled = min_max_scaler.fit_transform(x.astype(float))
crime_anal_hw2 = pd_hw2.DataFrame(x_scaled, columns = col, index = crime_anal.index)

col2 = ['강간검거율', '강도검거율', '살인검거율', '절도검거율', '폭력검거율']
crime_anal_hw2[col2] = crime_anal[col2]
crime_anal_hw2.head()
```

Out [19]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619

In [20]:
```
# data를 pandas로 읽어서 result_CCTV 변수에 저장합니다.
# .read_csv : CSV 파일을 읽는 명령, encoding = 'euc-kr' : 인코딩 data를 euc-kr을 사용하겠다
# euc-kr : 2바이트로 한글이나 일부 한자 등을 표현할 수 있게 만든 방식. 아스키값은 그대로 1바이트
# .head() : 인자값만큼 행을 출력한다. 인자를 default로 주면 pandas data의 첫 5행만 출력
# 구별 인구수와 CCTV개수를 추가합니다.
result_CCTV = pd_hw2.read_csv('../data/01. CCTV_result.csv', encoding='UTF-8', index_col='구별')
crime_anal_hw2[['인구수', 'CCTV']] = result_CCTV[['인구수', '소계']]
crime_anal_hw2.head()
```

Out[20]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594	570500.0	2780
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855	453233.0	773
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222	330192.0	748
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157	525515.0	1496
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619	372164.0	707

In [21]:
```
# 각 범죄들을 합한 것을 새로 만든 범죄항목에 옮겼습니다.
col = ['강간','강도','살인','절도','폭력']
crime_anal_hw2['범죄'] = np_hw2.sum(crime_anal_hw2[col], axis=1)
crime_anal_hw2.head()
```

Out [21]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV	범죄
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594	570500.0	2780	4.472701
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855	453233.0	773	1.116551
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222	330192.0	748	1.494746
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157	525515.0	1496	2.613667
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619	372164.0	707	2.034438

In [22]:
```
# 각 검거율도 합한것을 새로 만든 검거 항목에 옮겼습니다.
col = ['강간검거율','강도검거율','살인검거율','절도검거율','폭력검거율']
crime_anal_hw2['검거'] = np_hw2.sum(crime_anal_hw2[col], axis=1)
crime_anal_hw2.head()
```

Out [22]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV	범죄	검거
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594	570500.0	2780	4.472701	369.707384
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855	453233.0	773	1.116551	370.084431
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222	330192.0	748	1.494746	406.943540
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157	525515.0	1496	2.613667	368.622261
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619	372164.0	707	2.034438	416.915211

In [23]:
```
# 전체 데이터를 조회했습니다.
crime_anal_hw2
```

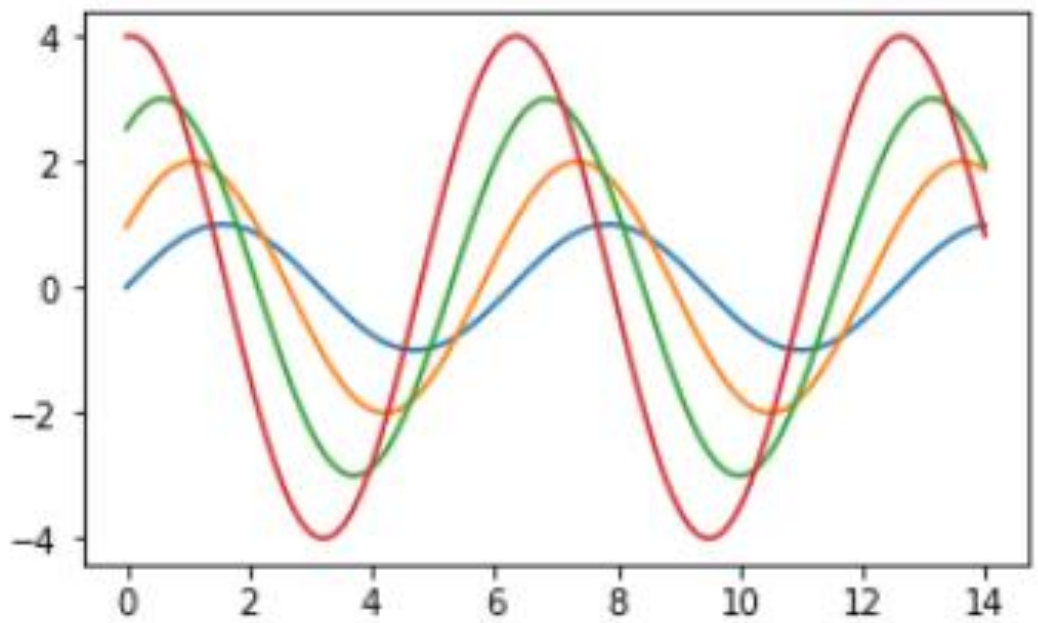
Out [23]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV	범죄	검거
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594	570500.0	2780	4.472701	369.707384
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855	453233.0	773	1.116551	370.084431
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222	330192.0	748	1.494746	406.943540
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157	525515.0	1496	2.613667	368.622261
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619	372164.0	707	2.034438	416.915211
구로구	0.515850	0.588235	0.500000	0.435169	0.359423	58.362989	73.333333	75.000000	38.072805	80.877951	447874.0	1561	2.398678	325.647079
금천구	0.141210	0.058824	0.083333	0.172426	0.134074	80.794702	100.000000	100.000000	56.668794	86.465433	255082.0	1015	0.589867	423.928929
노원구	0.273775	0.117647	0.666667	0.386589	0.292268	61.421320	100.000000	100.000000	36.525308	85.530665	569384.0	1265	1.736946	383.477292
도봉구	0.000000	0.235294	0.083333	0.000000	0.000000	100.000000	100.000000	100.000000	44.967074	87.626093	348646.0	485	0.318627	432.593167
동대문구	0.204611	0.470588	0.250000	0.314061	0.250887	84.393064	100.000000	100.000000	41.090358	87.401884	369496.0	1294	1.490147	412.885306
동작구	0.527378	0.235294	0.250000	0.274376	0.100024	48.771930	55.555556	100.000000	35.442359	83.089005	412520.0	1091	1.387071	322.858850
마포구	0.553314	0.529412	0.500000	0.510434	0.353748	84.013605	71.428571	100.000000	31.819961	84.445189	389649.0	574	2.446908	371.707327
서대문구	0.149856	0.000000	0.000000	0.256244	0.134547	80.519481	80.000000	100.000000	40.728477	83.219844	327163.0	962	0.540647	384.467802
서초구	0.838617	0.235294	0.500000	0.537804	0.215654	63.358779	66.666667	75.000000	41.404175	87.453105	450310.0	1930	2.327368	333.882725
성동구	0.069164	0.235294	0.166667	0.186110	0.029558	94.444444	88.888889	100.000000	37.149969	86.538462	311244.0	1062	0.686793	407.021764
성북구	0.138329	0.000000	0.250000	0.247007	0.170726	82.666667	80.000000	100.000000	41.512605	83.974649	461260.0	1464	0.806061	388.153921
송파구	0.340058	0.470588	0.750000	0.744441	0.427524	80.909091	76.923077	90.909091	34.856437	84.552352	667483.0	618	2.732611	368.150048
양천구	0.806916	0.823529	0.666667	1.000000	1.000000	77.486911	84.210526	100.000000	48.469644	83.065080	479978.0	2034	4.297113	393.232162
영등포구	0.556196	1.000000	1.000000	0.650359	0.493024	62.033898	90.909091	85.714286	32.995951	82.894737	402985.0	904	3.699580	354.547963
용산구	0.265130	0.529412	0.250000	0.169004	0.133128	89.175258	100.000000	100.000000	37.700706	83.121951	244203.0	1624	1.346674	409.997915
은평구	0.184438	0.235294	0.083333	0.291139	0.275715	84.939759	66.666667	100.000000	37.147335	86.920467	494388.0	1873	1.069920	375.674229
종로구	0.314121	0.352941	0.333333	0.383510	0.190589	76.303318	81.818182	83.333333	38.324176	84.212822	162820.0	1002	1.574494	363.991830
중구	0.195965	0.235294	0.083333	0.508040	0.174273	65.294118	66.666667	66.666667	33.712716	88.309353	133240.0	671	1.196905	320.649519
중랑구	0.244957	0.352941	0.916667	0.366746	0.321589	79.144385	81.818182	92.307692	38.829040	84.545135	414503.0	660	2.202900	376.644434

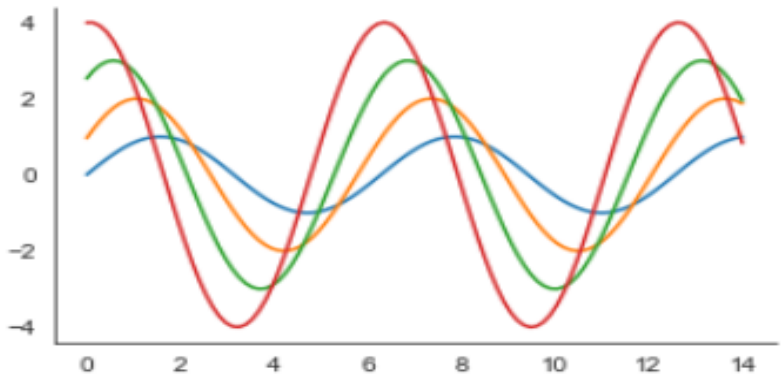
4. Seaborn을 이용한 시각화

In [24]:

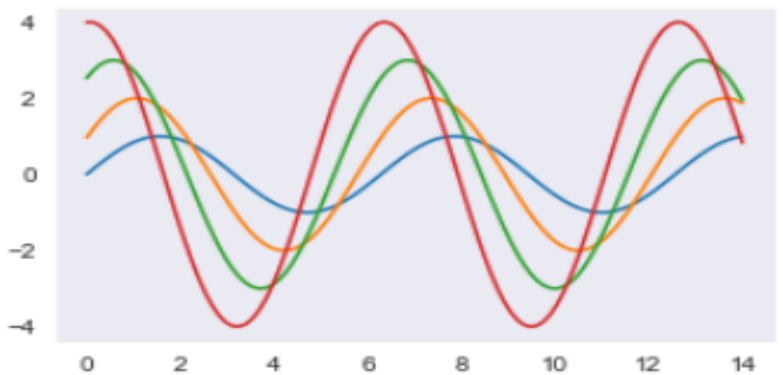
```
# pip install seaborn
# cmd에서 pip list > 설치된 모듈들 확인
# seaborn모듈을 import할때는
# matplotlib모듈을 먼저 import 해야합니다.
import matplotlib.pyplot as plt_hw2
%matplotlib inline

import seaborn as sns_hw2

x = np_hw2.linspace(0, 14, 100)
y1 = np_hw2.sin(x)
y2 = 2*np_hw2.sin(x+0.5)
y3 = 3*np_hw2.sin(x+1.0)
y4 = 4*np_hw2.sin(x+1.5)

plt_hw2.figure(figsize=(5,3))
plt_hw2.plot(x,y1, x,y2, x,y3, x,y4)
plt_hw2.show()
```



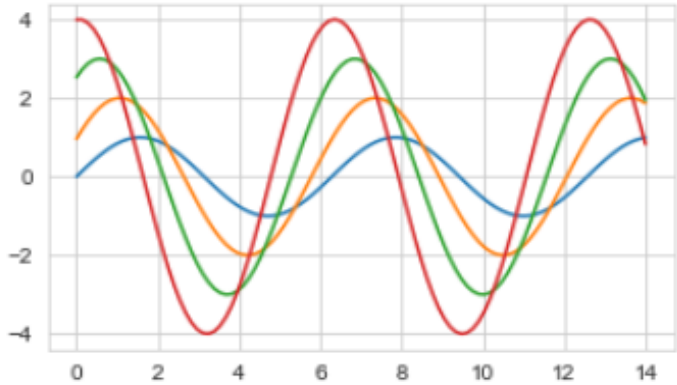
In [25]:
```
# 예제
sns_hw2.set_style("white")

plt_hw2.figure(figsize=(5,3))
plt_hw2.plot(x,y1, x,y2, x,y3, x,y4)

sns_hw2.despine()

plt_hw2.show()
```



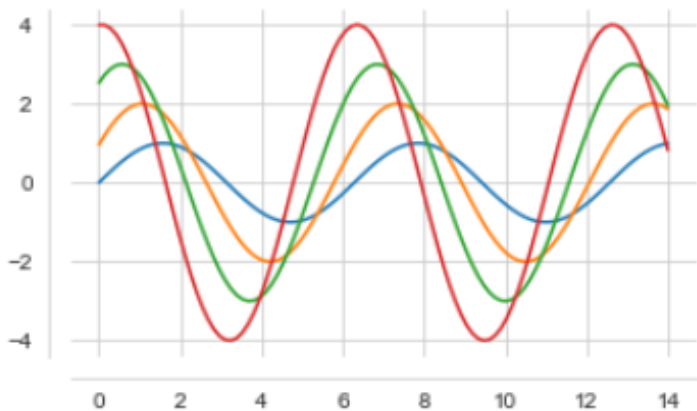
In [26]:
```
# 예제
sns_hw2.set_style("dark")

plt_hw2.figure(figsize=(5,3))
plt_hw2.plot(x,y1, x,y2, x,y3, x,y4)
plt_hw2.show()
```



In [27]:
```
# seaborn은 set_style을 사용해 스타일을 조절합니다.
sns_hw2.set_style("whitegrid")

plt_hw2.figure(figsize=(5,3))
plt_hw2.plot(x,y1, x,y2, x,y3, x,y4)
plt_hw2.show()
```



In [28]:
```
# 예제
plt_hw2.figure(figsize=(5,3))
plt_hw2.plot(x,y1, x,y2, x,y3, x,y4)

sns_hw2.despine(offset=10)

plt_hw2.show()
```



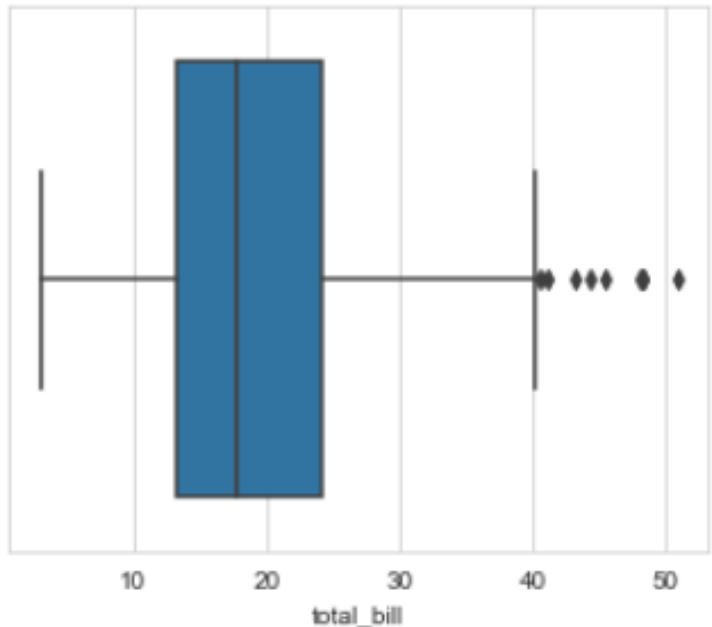
In [29]:
```
# %matplotlib inline : notebook을 실행한 브라우저에서
# 바로 그림을 볼 수 있게 해줍니다.
sns_hw2.set_style("whitegrid")
%matplotlib inline
```

In [30]:
```
# tips 데이터셋에는 요일별 점심,저녁,흡연여부,
# 식사금액,팁이 정리되어 있습니다.
tips = sns_hw2.load_dataset("tips")
tips.head(5)
```

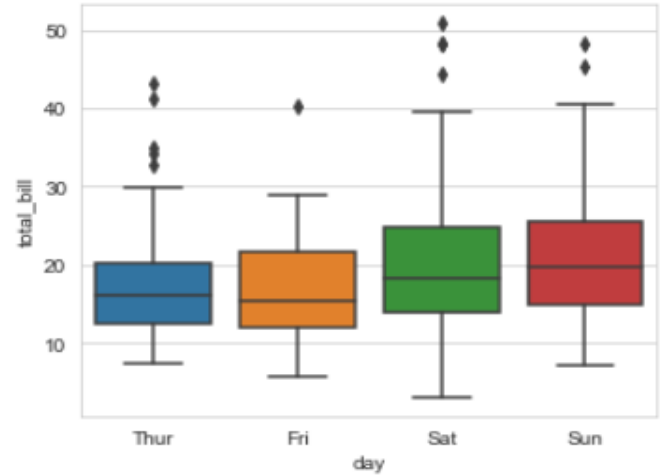
Out[30]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

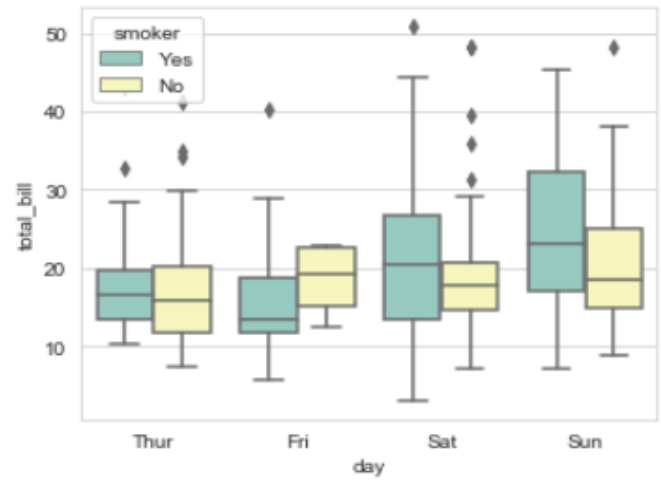
In [31]:
```
# 예제
sns_hw2.set_style("whitegrid")

plt_hw2.figure(figsize=(5,4))
sns_hw2.boxplot(x=tips["total_bill"])
plt_hw2.show()
```



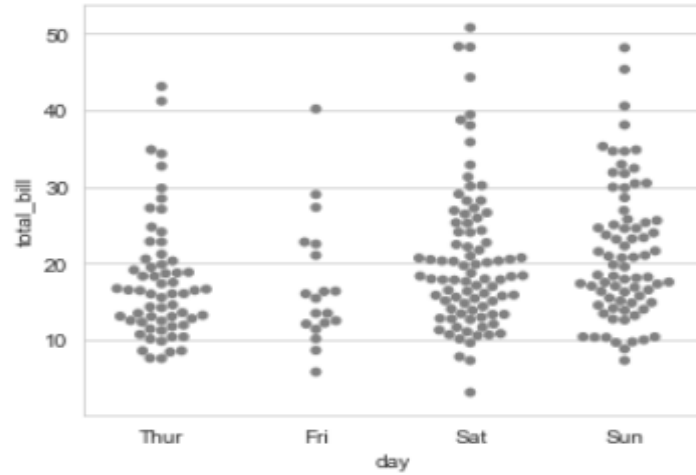
In [32]:
```
# .boxplot을 사용해 boxplot을 그립니다.
# x축은 요일, y축은 전체 금액을 그렸습니다.
plt_hw2.figure(figsize=(5,4))
sns_hw2.boxplot(x="day", y="total_bill", data=tips)
plt_hw2.show()
```



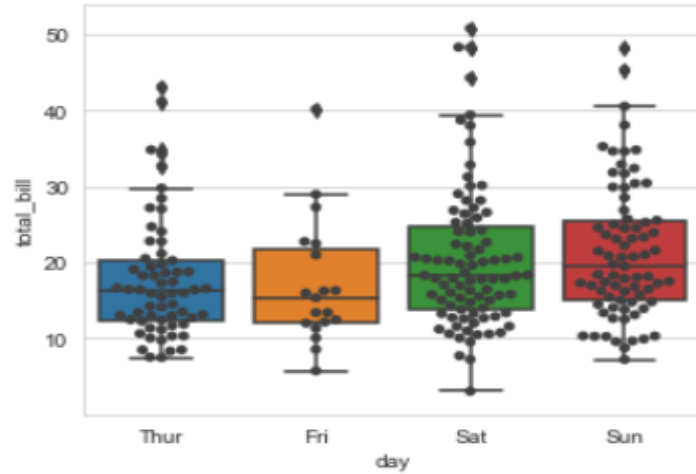
In [33]:
```
# hue 옵션을 사용해서 구분이 가능합니다.
# 흡연 여부로 구분했습니다.
plt_hw2.figure(figsize=(5,4))
sns_hw2.boxplot(x="day", y="total_bill", hue="smoker"
                , data=tips, palette="Set3")
plt_hw2.show()
```



In [34]:
```
# 예제
plt_hw2.figure(figsize=(5,4))
sns_hw2.swarmplot(x="day", y="total_bill",
                  data=tips, color=".5")
plt_hw2.show()
```

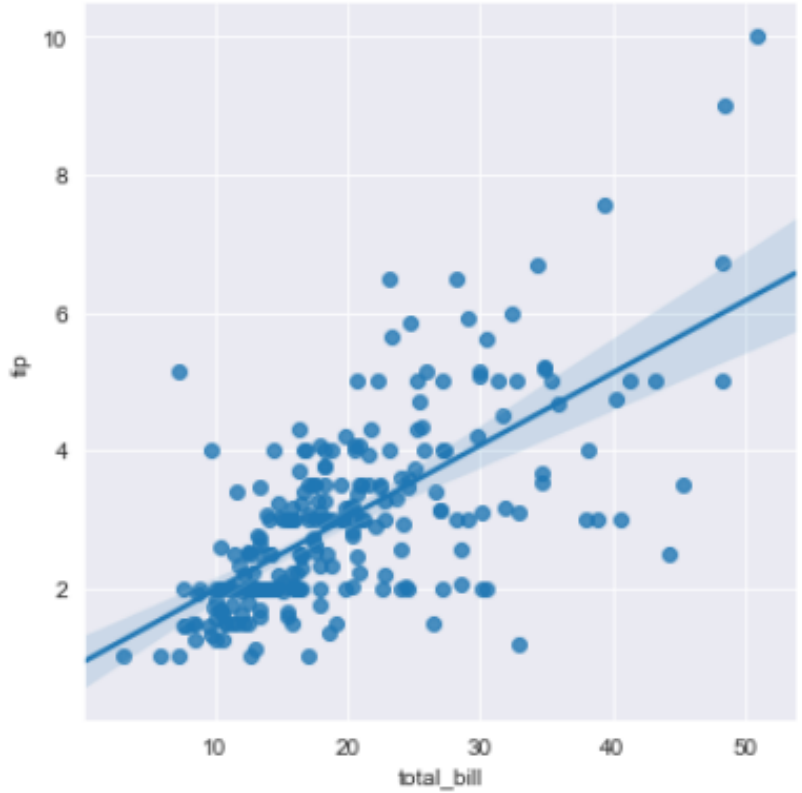


In [35]:
```
# 예제
plt_hw2.figure(figsize=(5,4))
sns_hw2.boxplot(x="day", y="total_bill", data=tips)
sns_hw2.swarmplot(x="day", y="total_bill",
                  data=tips, color=".25")
plt_hw2.show()
```



In [37]:
```
# 직선을 darkgrid 스타일로 implot을 그립니다
sns_hw2.set_style("darkgrid")
sns_hw2.lmplot(x="total_bill", y="tip", data=tips, size=5)
plt_hw2.show()
```

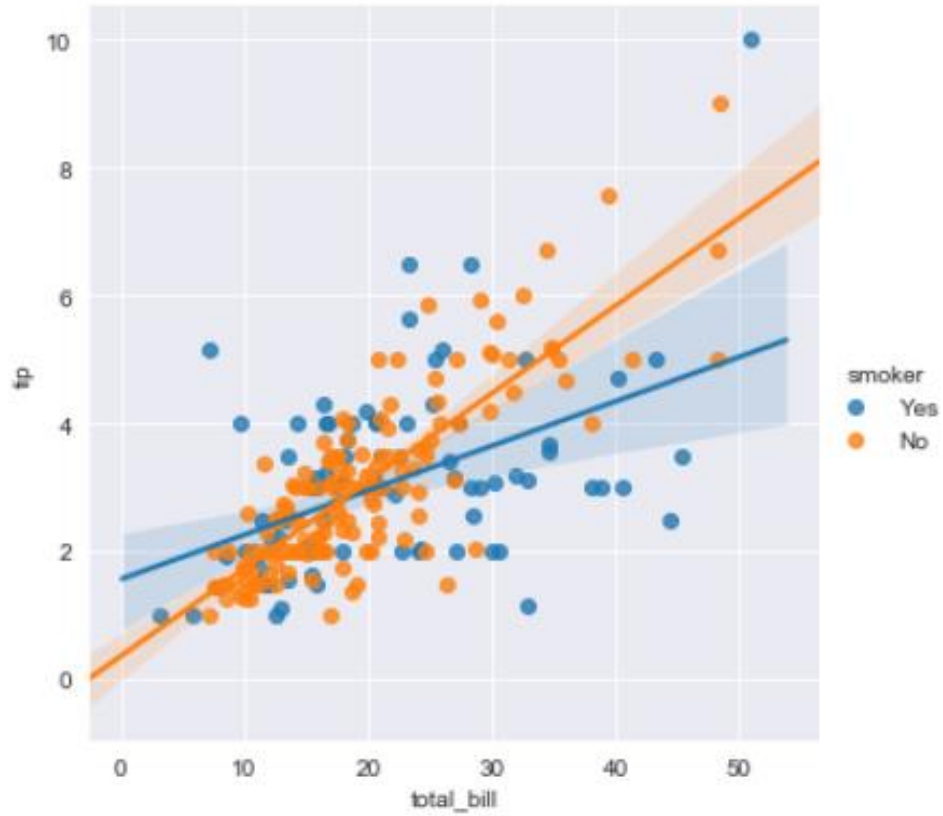
```
C:₩Users₩nerin₩Anaconda3₩lib₩site-packages₩seaborn₩regression.py:546: UserWarning: The `size` paramter has been renamed to `height`; p
lease update your code.
  warnings.warn(msg, UserWarning)
```



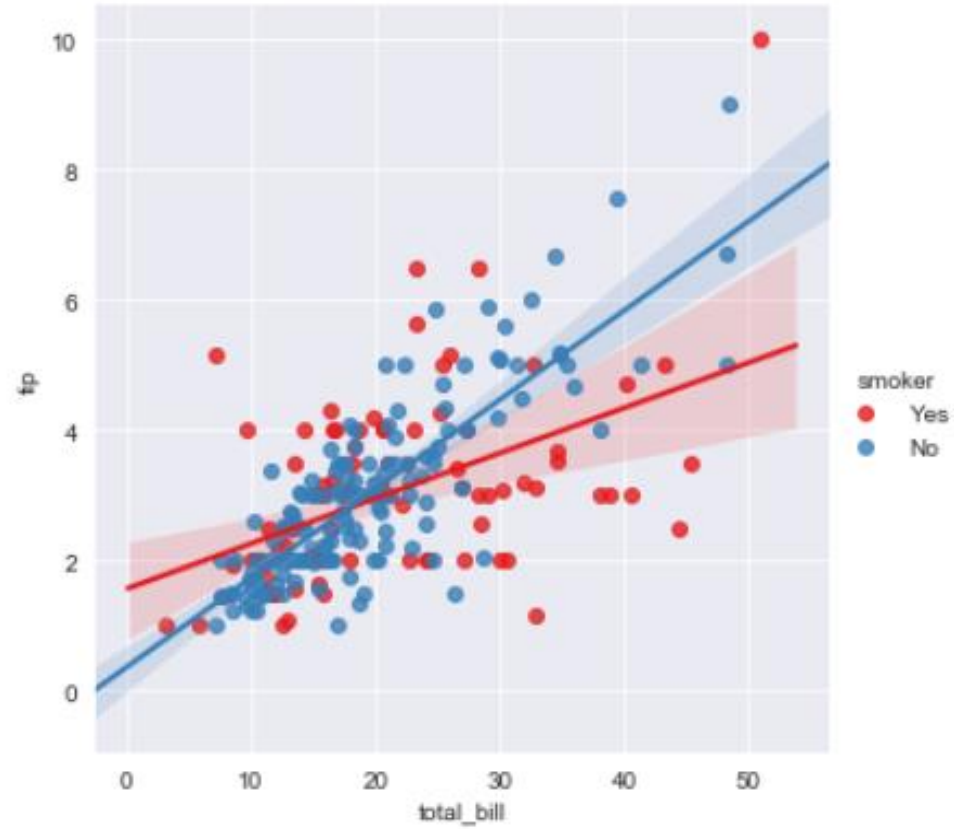
버전이 달라서 발생한 오류입니다.

미래의 파이썬 버전에서 지원되지 않을 수 있는 언어를 사용했다고 경고가 출력되었습니다.

In [38]:
```
# implot도 hue옵션을 사용해서 구분가능합니다.
sns_hw2.lmplot(x="total_bill", y="tip",
               hue="smoker", data=tips, size=5)
plt_hw2.show()
```



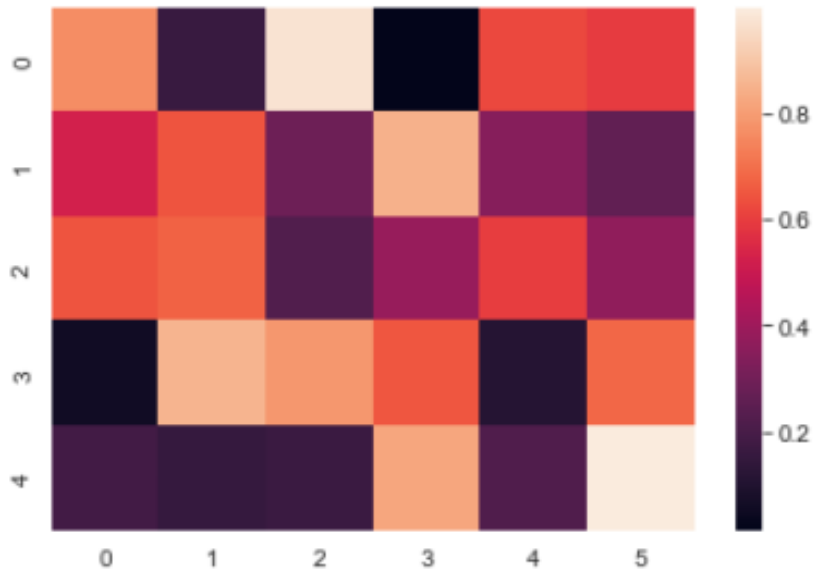
In [39]:
```
# palette로 색상을 지정할 수 있습니다.
sns_hw2.lmplot(x="total_bill", y="tip", hue="smoker"
               , data=tips, palette="Set1", size=5)
plt_hw2.show()
```



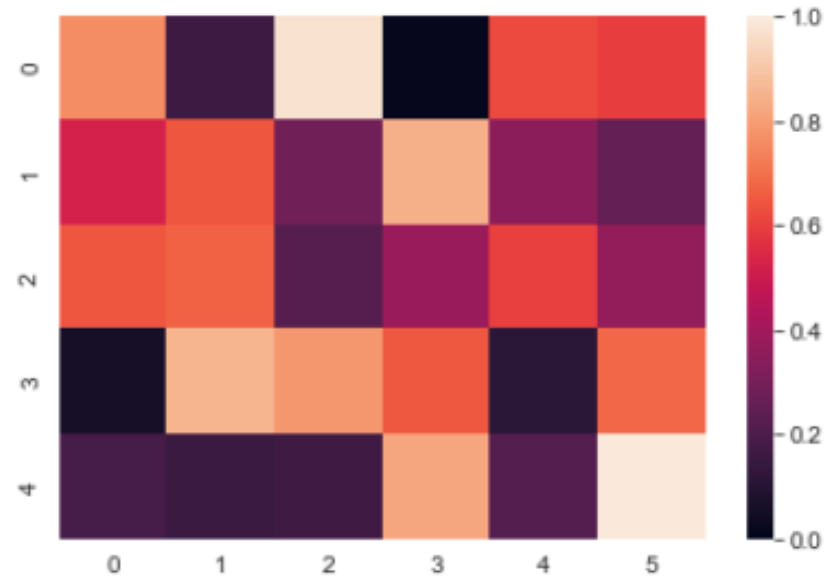
In [40]:
```
# heatmap에 사용할 데이터
# rand()함수를 사용해서 데이터를 무작위로 수집합니다.
uniform_data = np_hw2.random.rand(5, 6)
uniform_data
```

Out[40]:
```
array([[0.7632243 , 0.16082953, 0.97652482, 0.01472288, 0.62194358,
        0.59719317],
       [0.5251071 , 0.64704116, 0.28689523, 0.84657969, 0.34796001,
        0.26077159],
       [0.6446866 , 0.67074121, 0.22533966, 0.38410139, 0.60107527,
        0.36862889],
       [0.05889572, 0.85685551, 0.7875849 , 0.64943562, 0.11166692,
        0.68301325],
       [0.18596791, 0.15454403, 0.16772565, 0.82175282, 0.2216541 ,
        0.99604692]])
```

In [41]:
```
# heatmap 예제
sns_hw2.heatmap(uniform_data)
plt_hw2.show()
```



In [42]:
```
# heatmap 예제
sns_hw2.heatmap(uniform_data, vmin=0, vmax=1)
plt_hw2.show()
```



In [43]:
```
# 연도와 월별 항공기 승객수를 기록한 데이터
flights = sns_hw2.load_dataset("flights")
flights.head(5)
```

Out[43]:

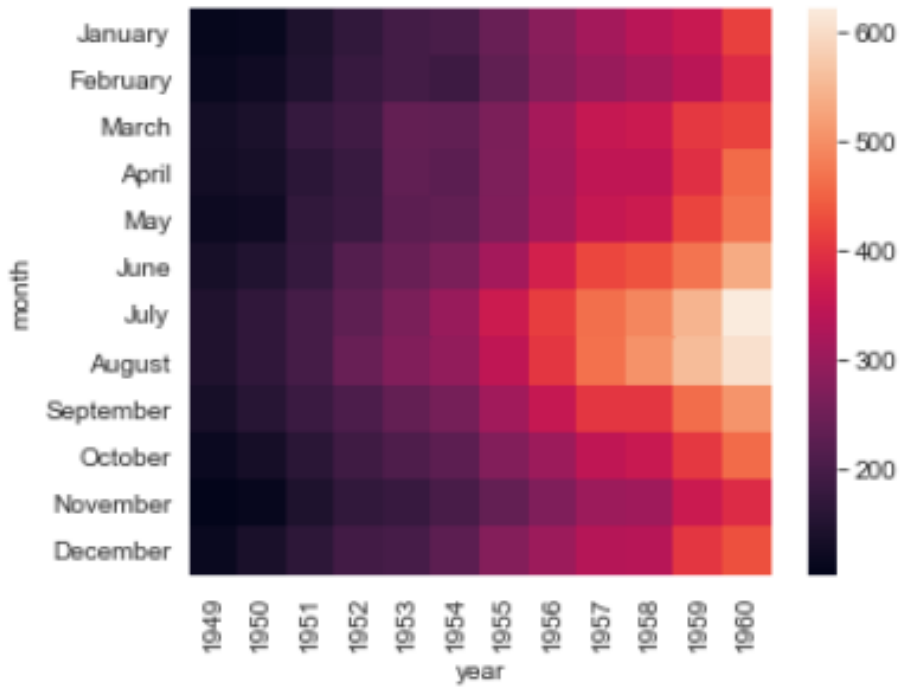
	year	month	passengers
0	1949	January	112
1	1949	February	118
2	1949	March	132
3	1949	April	129
4	1949	May	121

In [44]:
```
# .pivot으로 월, 년, 승객으로 구분했습니다.
flights = flights.pivot("month", "year", "passengers")
flights.head(5)
```

Out[44]:

year	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960
month												
January	112	115	145	171	196	204	242	284	315	340	360	417
February	118	126	150	180	196	188	233	277	301	318	342	391
March	132	141	178	193	236	235	267	317	356	362	406	419
April	129	135	163	181	235	227	269	313	348	348	396	461
May	121	125	172	183	229	234	270	318	355	363	420	472

In [45]:
```
# heatmap
plt_hw2.figure(figsize=(5,4))
sns_hw2.heatmap(flights)
plt_hw2.show()
```



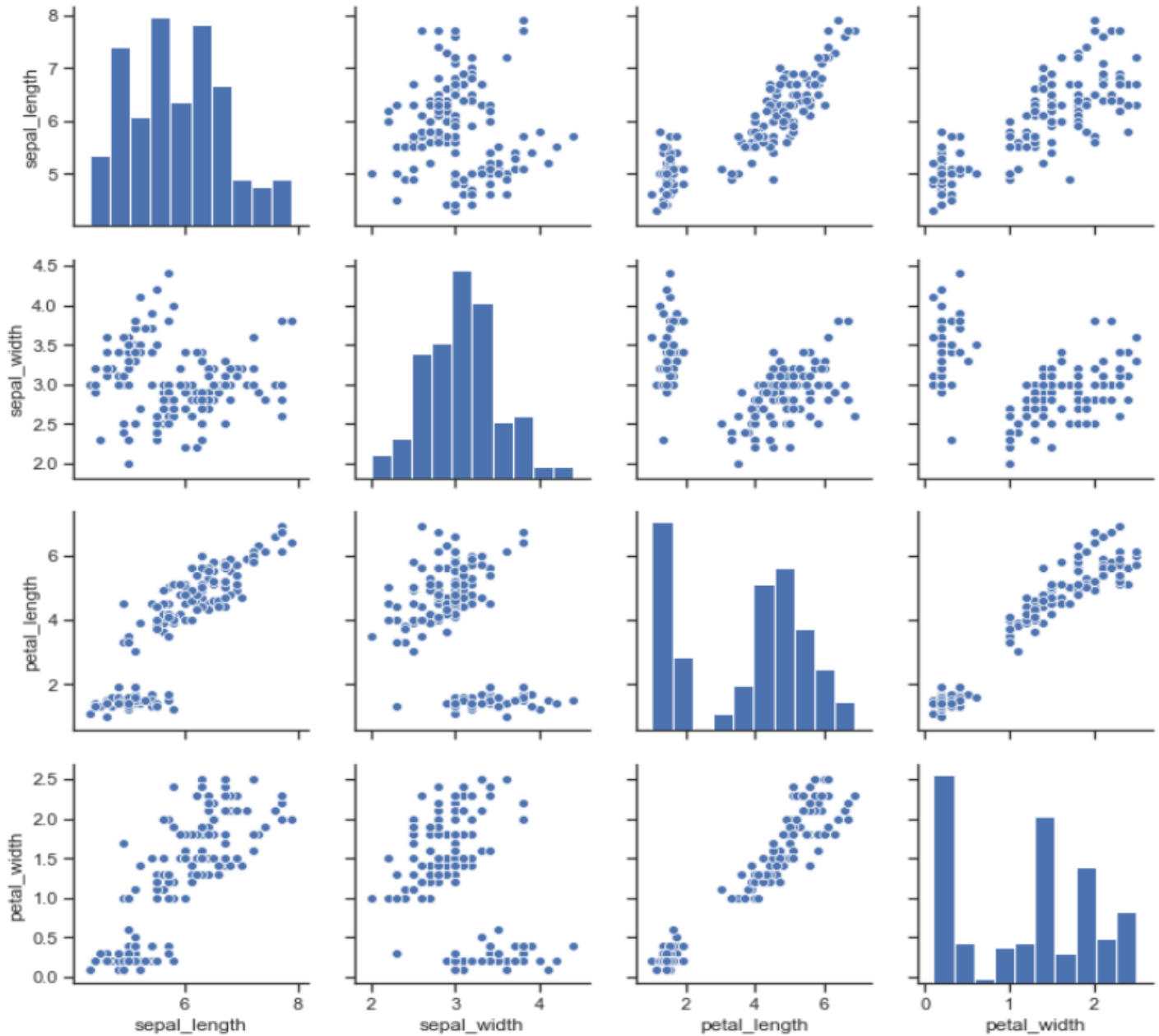
In [46]:
```
# heatmap
plt_hw2.figure(figsize=(5,4))
sns_hw2.heatmap(flights, annot=True, fmt="d")
plt_hw2.show()
```

In [47]:
```
# 아이리스 꽃에 대한 데이터입니다.
# 꽃잎의 너비, 꽃잎의 폭, 꽃받침의 너비, 꽃받침의 폭
sns_hw2.set(style="ticks")
iris = sns_hw2.load_dataset("iris")
iris.head(10)
```

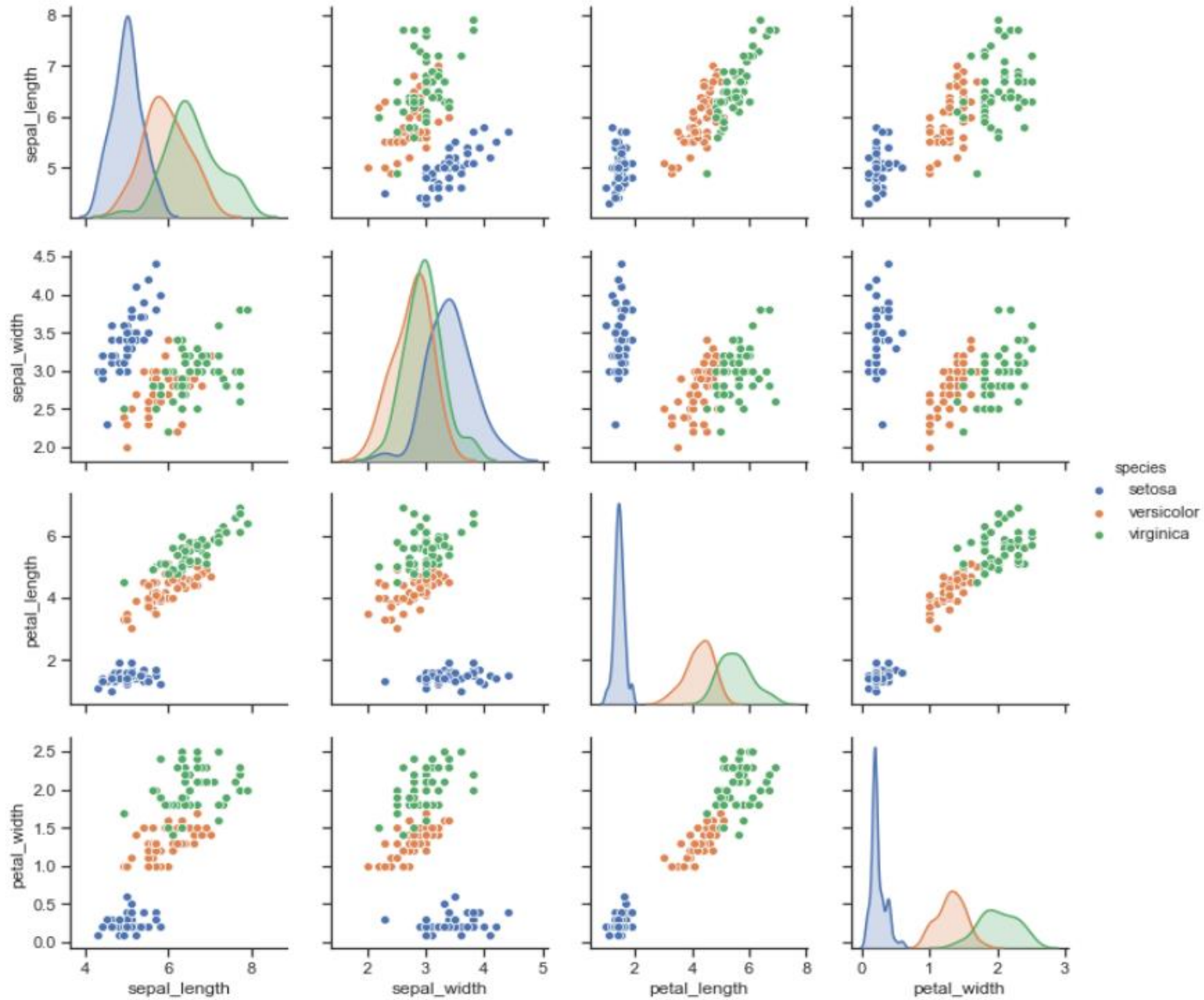
Out[47]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa

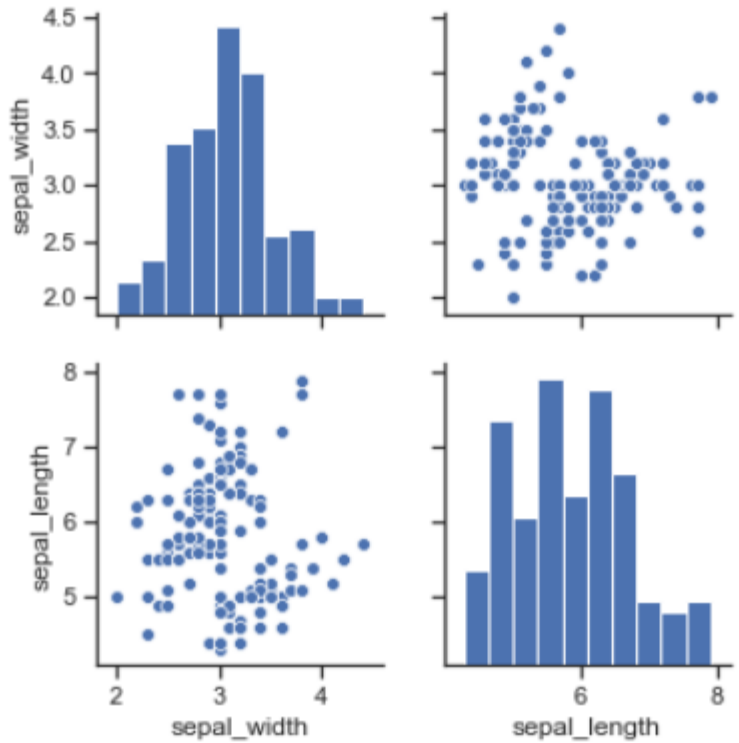
In [48]:
```
# pairplot을 사용하면
# 데이터에 대한 상관관계, 분류 특성 등을 한눈에 알 수 있습니다.
sns_hw2.pairplot(iris)
plt_hw2.show()
```



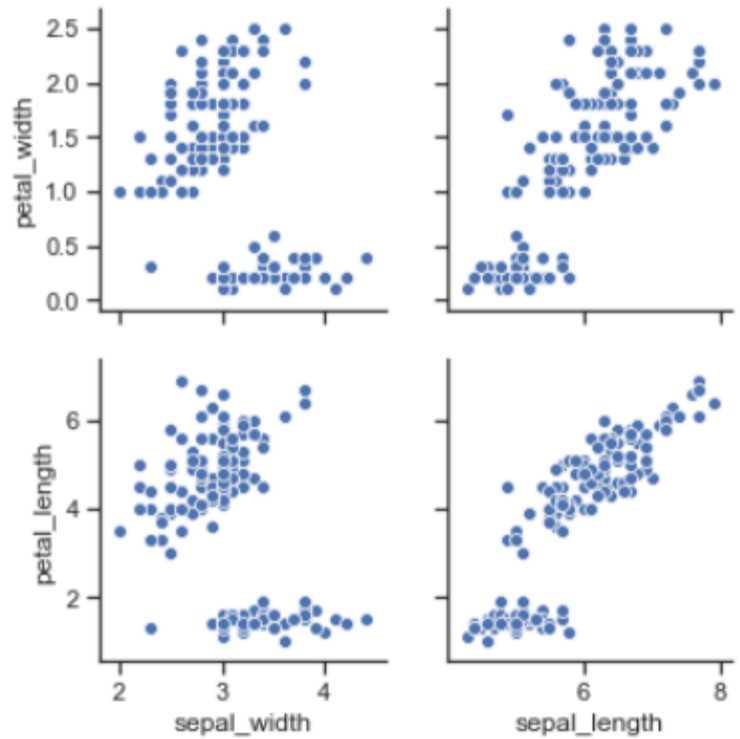
In [49]:
```
# hue 옵션을 사용해서 구분할수도 있습니다.
sns_hw2.pairplot(iris, hue="species")
plt_hw2.show()
```



In [50]:
```
# 열을 골라서 볼 수 있습니다.
sns_hw2.pairplot(iris, vars=["sepal_width", "sepal_length"])
plt_hw2.show()
```



In [51]:
```
# 예제
sns_hw2.pairplot(iris, x_vars=["sepal_width", "sepal_length"],
                 y_vars=["petal_width", "petal_length"])
plt_hw2.show()
```

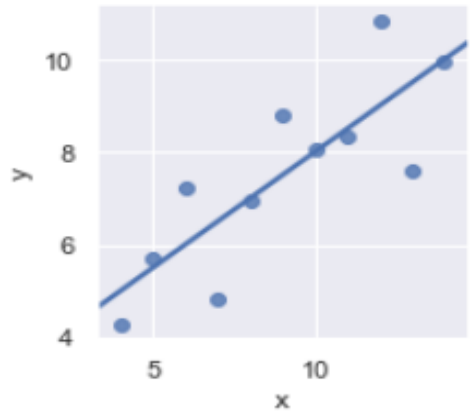


In [52]:
```
# 예제
anscombe = sns_hw2.load_dataset("anscombe")
anscombe.head(5)
```

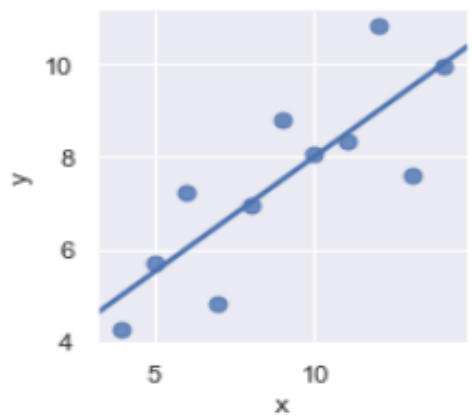
Out[52]:

	dataset	x	y
0	I	10.0	8.04
1	I	8.0	6.95
2	I	13.0	7.58
3	I	9.0	8.81
4	I	11.0	8.33

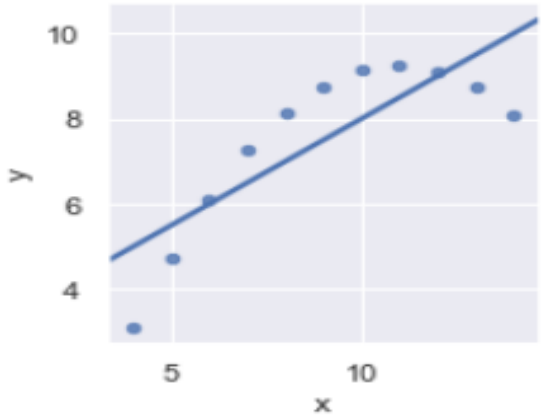
In [53]:
```
# 예제
sns_hw2.set_style("darkgrid")

sns_hw2.lmplot(x="x", y="y", data=anscombe.query("dataset == 'I'"),  ci=None, size=3)
plt_hw2.show()
```



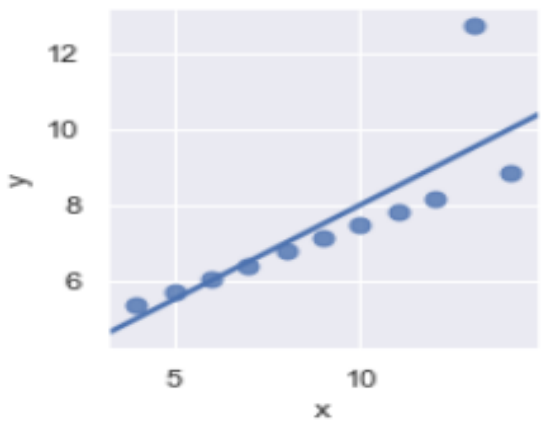
In [54]:
```
# 예제

sns_hw2.lmplot(x="x", y="y", data=anscombe.query("dataset == 'I'"),
           ci=None, scatter_kws={"s": 50}, size=3)
plt_hw2.show()
```



In [55]:
```
# 예제
sns_hw2.lmplot(x="x", y="y", data=anscombe.query("dataset == 'II'"),
               order=1, ci=None, scatter_kws={"s": 20}, size=3)
plt_hw2.show()
```



In [56]:
```
# 예제

sns_hw2.lmplot(x="x", y="y", data=anscombe.query("dataset == 'III'"),
               ci=None, scatter_kws={"s": 50}, size=3)
plt_hw2.show()
```



5. 범죄데이터 시각화

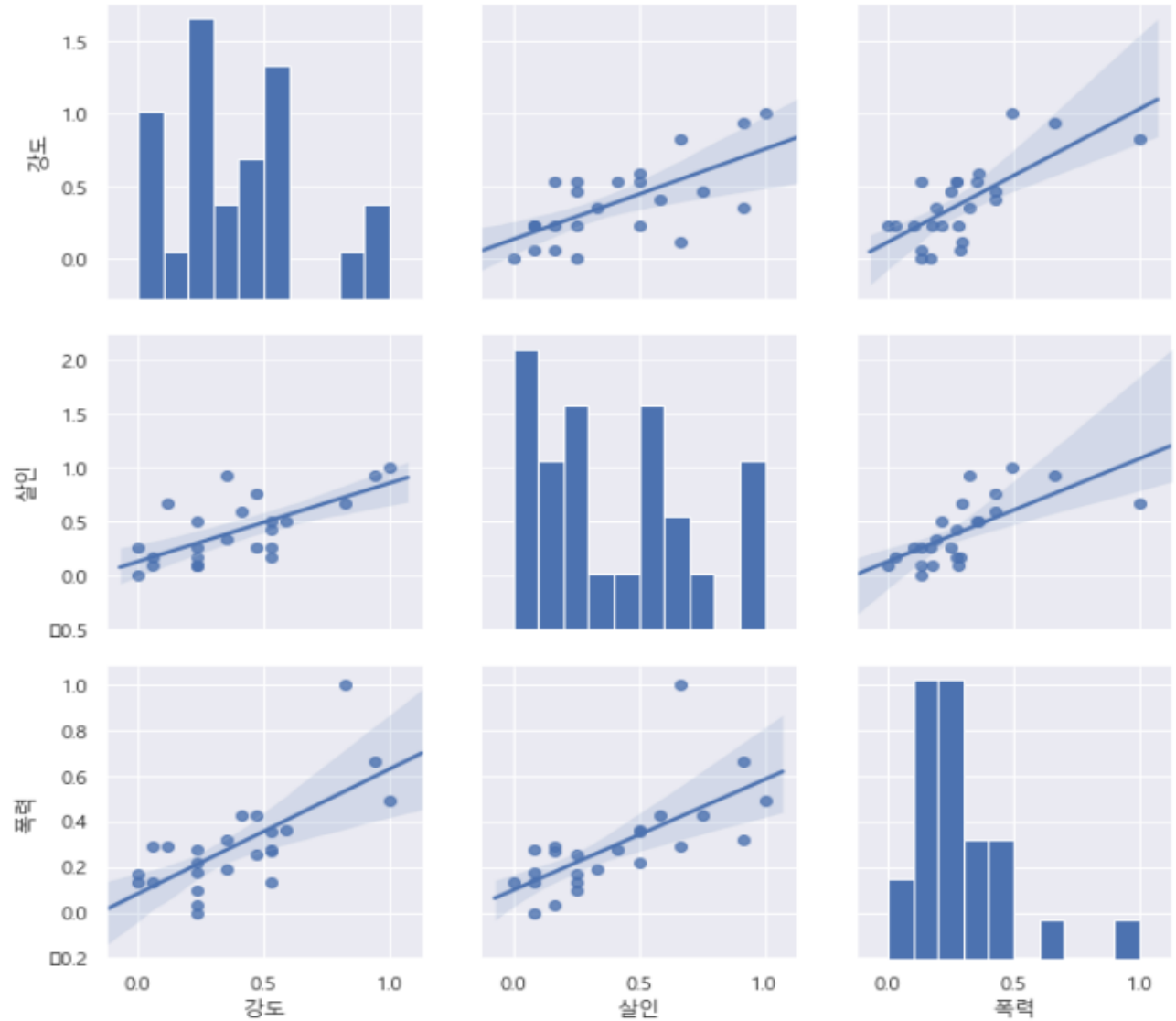
In [57]:
```
# 한글폰트문제를 해결합니다
# 이쁜 폰트 잘 안됀다. 막아놓은 경우가 많으니 에러나면 다른거 사용!
%matplotlib inline

import platform
path = "c:/Windows/Fonts/malgun.ttf"
from matplotlib import font_manager, rc
if platform.system() == 'Darwin':
    rc('font', family='AppleGothic')
elif platform.system() == 'Windows':
    font_name = font_manager.FontProperties(fname=path).get_name()
    rc('font', family=font_name)
else:
    print('Unknown system... sorry~~~~')
```

In [58]:
```
# 데이터셋의 상위 5행만 출력합니다.
crime_anal_hw2.head()
```

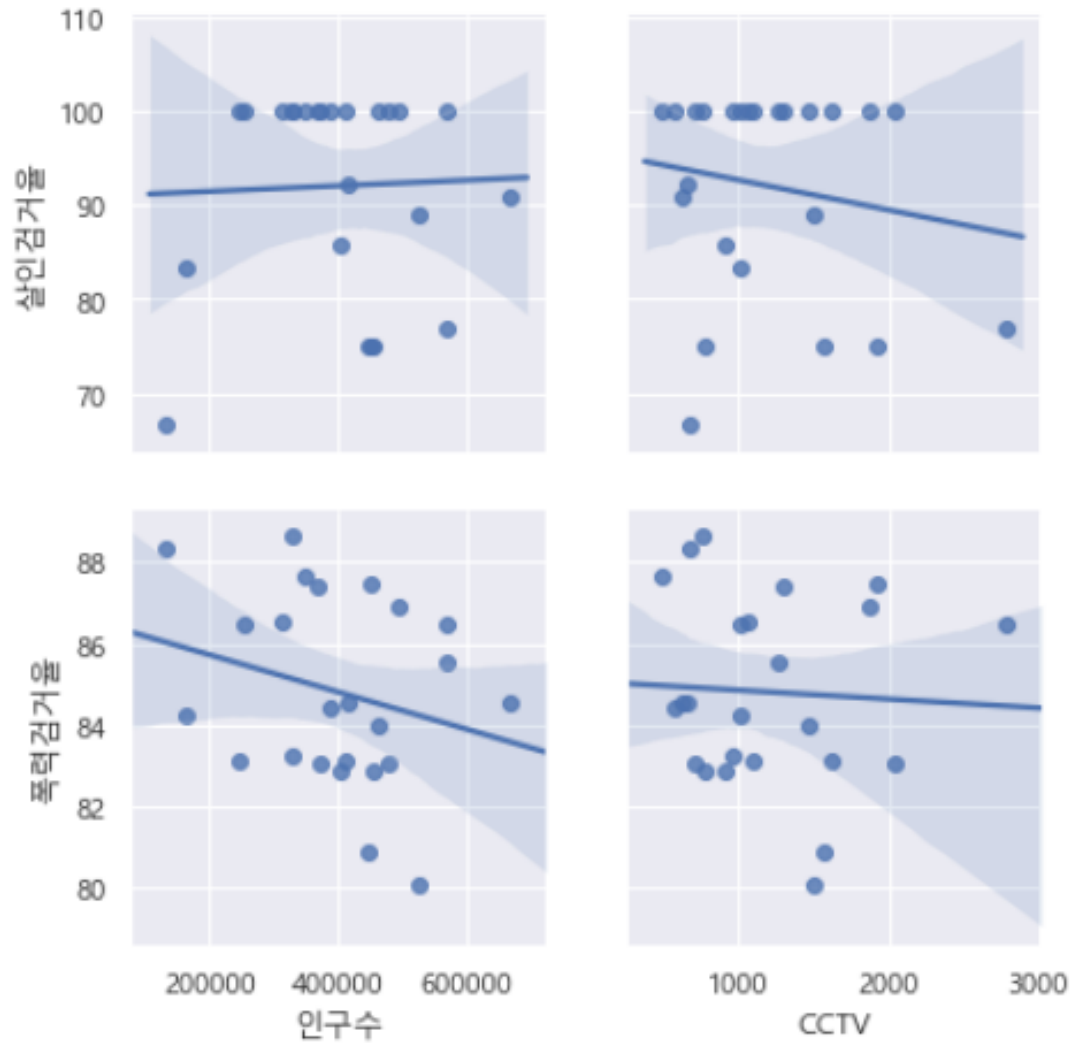
Out[58]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV	범죄	검거
강남구	1.000000	0.941176	0.916667	0.953472	0.661386	77.728285	85.714286	76.923077	42.857143	86.484594	570500.0	2780	4.472701	369.707384
강동구	0.155620	0.058824	0.166667	0.445775	0.289667	78.846154	100.000000	75.000000	33.347422	82.890855	453233.0	773	1.116551	370.084431
강북구	0.146974	0.529412	0.416667	0.126924	0.274769	82.352941	92.857143	100.000000	43.096234	88.637222	330192.0	748	1.494746	406.943540
관악구	0.628242	0.411765	0.583333	0.562094	0.428234	69.062500	100.000000	88.888889	30.561715	80.109157	525515.0	1496	2.613667	368.622261
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.000000	100.000000	42.200925	83.047619	372164.0	707	2.034438	416.915211

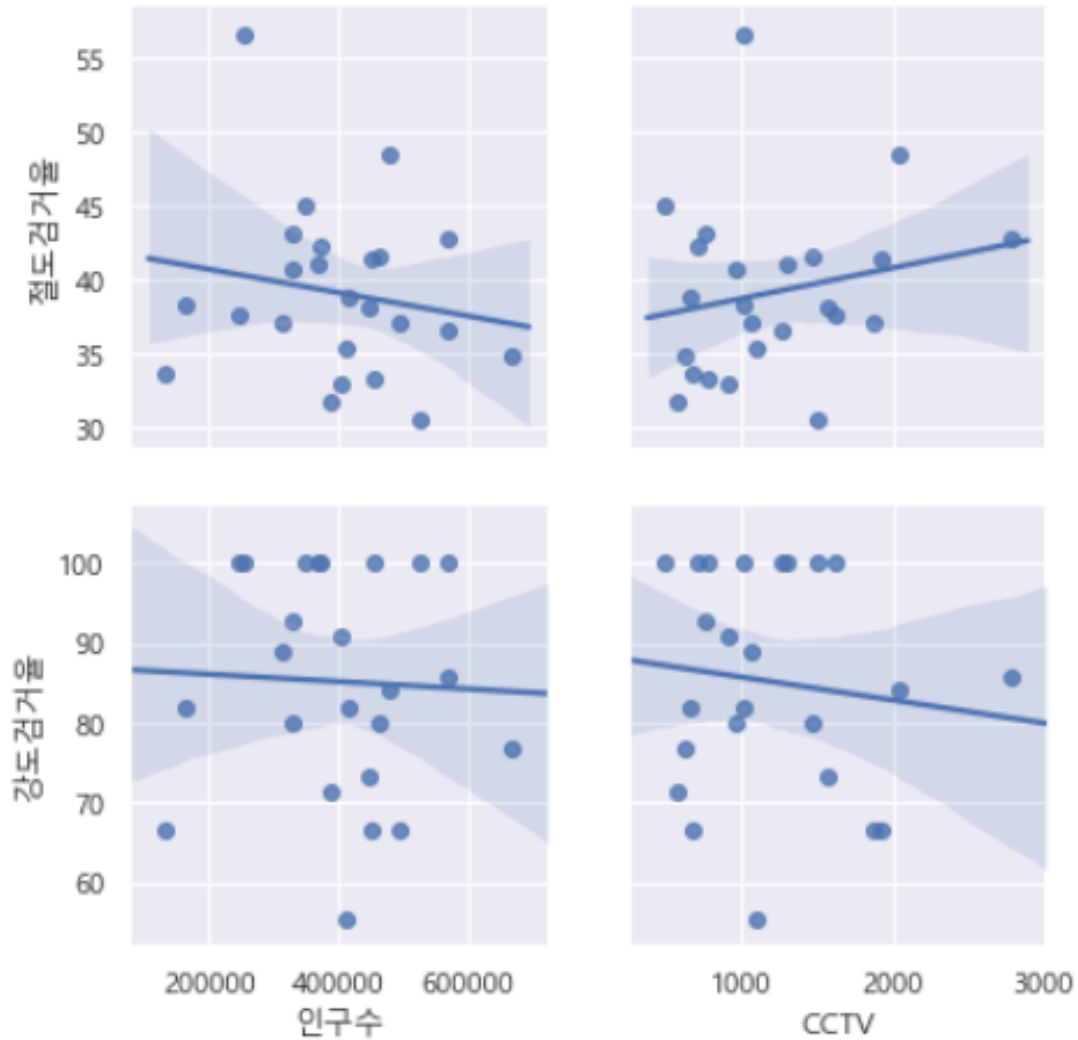
In [59]:
```
# 강도, 살인, 폭력에 대해 그래프로 표현합니다.
sns_hw2.pairplot(crime_anal_hw2, vars=["강도", "살인", "폭력"], kind='reg', size=3)
plt_hw2.show()
```



In [60]:
```
# 인구수와 CCTV, 살인검거율과 폭력검거율에 대해 그래프로 표현합니다.
sns_hw2.pairplot(crime_anal_hw2, x_vars=["인구수", "CCTV"],
                 y_vars=["살인검거율", "폭력검거율"], kind='reg', size=3)
plt_hw2.show()
```



In [61]:
```
# 인구수와 CCTV, 절도검거율과 강도검거율에 대해 그래프로 표현합니다.
sns_hw2.pairplot(crime_anal_hw2, x_vars=["인구수", "CCTV"],
                 y_vars=["절도검거율", "강도검거율"], kind='reg', size=3)
plt_hw2.show()
```

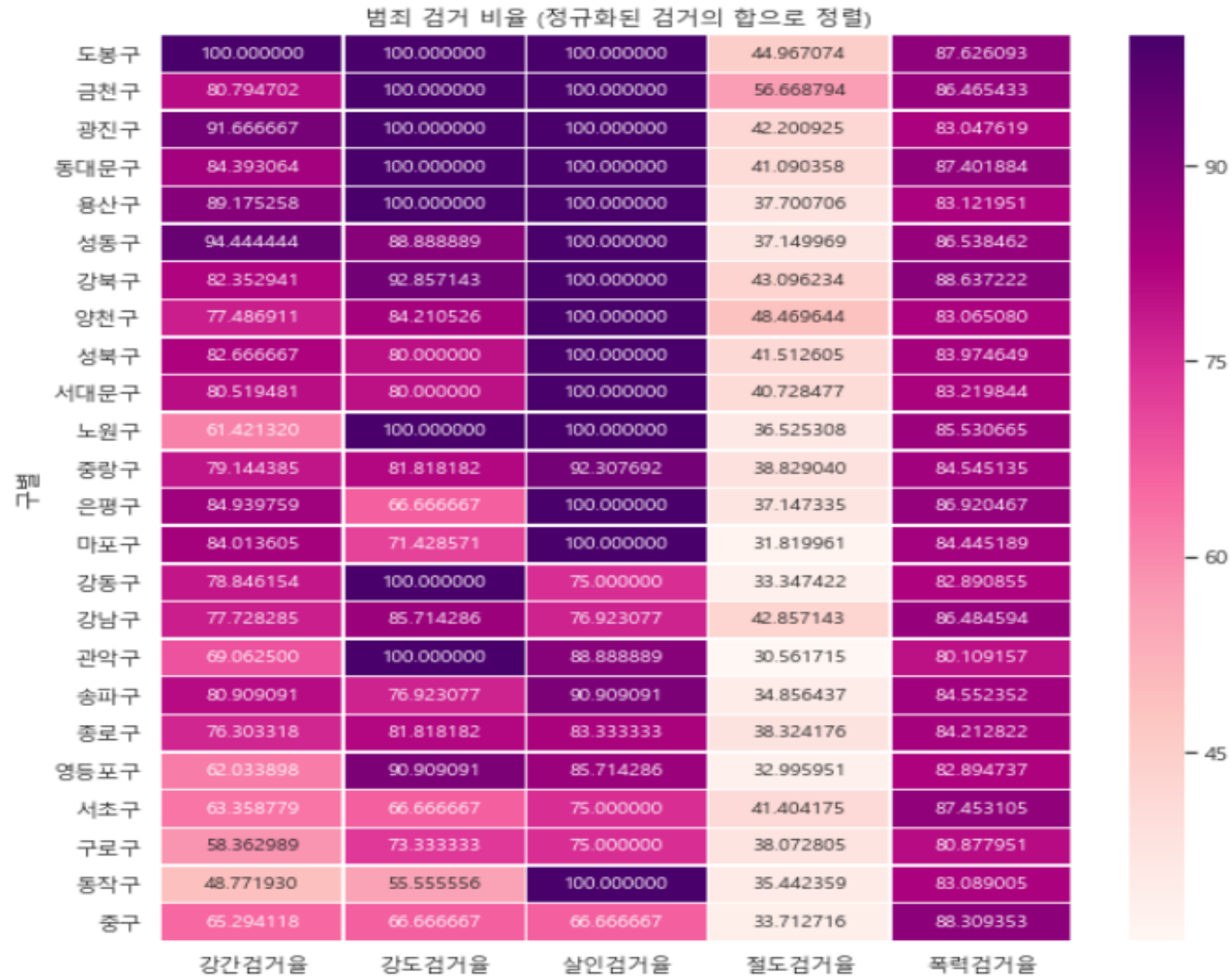


In [62]:
```
# 검거율의 최대값을 100으로 한정해서 내림차순으로 정렬했습니다.
tmp_max = crime_anal_hw2['검거'].max()
crime_anal_hw2['검거'] = crime_anal_hw2['검거'] / tmp_max * 100
crime_anal_norm_sort = crime_anal_hw2.sort_values(by='검거', ascending=False)
crime_anal_norm_sort.head()
```

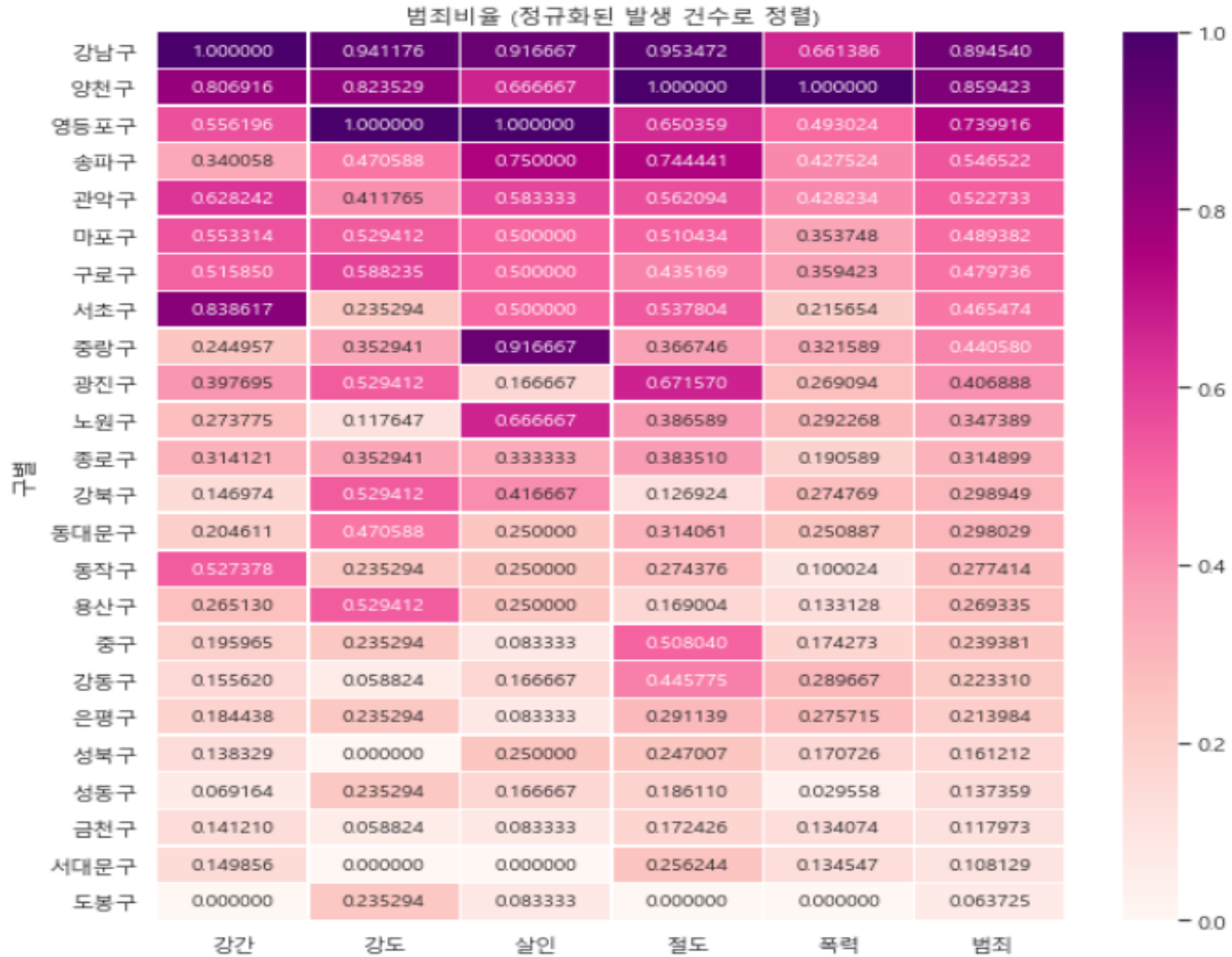
Out [62]:

구별	강간	강도	살인	절도	폭력	강간검거율	강도검거율	살인검거율	절도검거율	폭력검거율	인구수	CCTV	범죄	검거
도봉구	0.000000	0.235294	0.083333	0.000000	0.000000	100.000000	100.0	100.0	44.967074	87.626093	348646.0	485	0.318627	100.000000
금천구	0.141210	0.058824	0.083333	0.172426	0.134074	80.794702	100.0	100.0	56.668794	86.465433	255082.0	1015	0.589867	97.997139
광진구	0.397695	0.529412	0.166667	0.671570	0.269094	91.666667	100.0	100.0	42.200925	83.047619	372164.0	707	2.034438	96.375820
동대문구	0.204611	0.470588	0.250000	0.314061	0.250887	84.393064	100.0	100.0	41.090358	87.401884	369496.0	1294	1.490147	95.444250
용산구	0.265130	0.529412	0.250000	0.169004	0.133128	89.175258	100.0	100.0	37.700706	83.121951	244203.0	1624	1.346674	94.776790

In [63]:
```
# 검거율의 합으로 정렬한 데이터를 사용해 heatmap을 그립니다.
target_col = ['강간검거율', '강도검거율', '살인검거율', '절도검거율', '폭력검거율']

crime_anal_norm_sort = crime_anal_hw2.sort_values(by='검거', ascending=False)

plt_hw2.figure(figsize = (10,10))
sns_hw2.heatmap(crime_anal_norm_sort[target_col], annot=True, fmt='f',
                linewidths=.5, cmap='RdPu')
plt_hw2.title('범죄 검거 비율 (정규화된 검거의 합으로 정렬)')
plt_hw2.show()
```



In [64]:
```
# 범죄 발생 건수의 합으로 정렬한 데이터를 사용해 heatmap을 그립니다.
target_col = ['강간', '강도', '살인', '절도', '폭력', '범죄']

crime_anal_hw2['범죄'] = crime_anal_hw2['범죄'] / 5
crime_anal_norm_sort = crime_anal_hw2.sort_values(by='범죄', ascending=False)

plt_hw2.figure(figsize = (10,10))
sns_hw2.heatmap(crime_anal_norm_sort[target_col], annot=True, fmt='f', linewidths=.5,
                cmap='RdPu')
plt_hw2.title('범죄비율 (정규화된 발생 건수로 정렬)')
plt_hw2.show()
```



In [65]:
```
# 파일 저장
crime_anal_hw2.to_csv('../data/02. crime_in_Seoul_final.csv', sep=',',
                      encoding='utf-8')
```

9. 소감

이번 과제는 1장과 비슷해서 편하게 과제를 할 수 있었습니다.

그리고 무엇보다 교수님께서 과제 범위를 줄여주셔서
예상했던 시간보다 빨리 끝나서 좋았습니다.

아쉬웠던 점은 빨리 끝냈다곤 하지만.. 역시 시간이 부족해서 구글 맵 이용해서 데이터를 얻는 부분을 넘긴것입니다. 나중에 만들 기말 프로젝트에 꼭 넣어보고 싶다고 생각했습니다.

프로젝트 말하니까 이대로면 살짝 걱정이 되기도 합니다.
과제할때는 좀 더 꼼꼼하게 주석달아가면서 공부하는걸 목표로 해야 할 것 같습니다.